PRESEK
List za mlade matematike, fizike, astronome in računalnikarje

ISSN 0351-6652
Letnik **29** (2001/2002)
Številka 4
Strani 210–213

Martin Juvan:

RAČUNANJE Z VELIKIMI ŠTEVILI – UBASIC

Ključne besede: računalništvo, programska oprema, velika števila, razstavljanje na prafaktorje.

Elektronska verzija: http://www.presek.si/29/1482-Juvan.pdf

RAČUNANJE Z VELIKIMI ŠTEVILI – UBASIC

V Preseku sem ter tja pišemo o nalogah, ki zahtevajo računanje z velikimi števili (z velikimi števili mislim na cela števila, ki imajo precej več kot deset števk). Večkrat nam reševanje močno olajša uporaba računalnika, opremljenega s primerno programsko opremo.

Ena od možnosti je, da uporabimo katerega od paketov za simbolično računanje. Sam tovrstne probleme običajno rešujem s pomočjo programa *Mathematica*. A to je ena od ugodnosti akademskega okolja, saj ta program za običajnega uporabnika ni prav poceni in je tako precej težko dostopen. Računanja se lahko lotimo tudi s programom *Derive*. Ta je sicer manj zmogljiv, a zato prijaznejši za uporabo. Za občasnega reševalca Presekovih nalog pa je njegova glavna prednost pred Mathematico ta, da je moč na naslovu `www.derive.com` dobiti v 30-dnevni preizkus neokrnjeno različico programa. In 30 dni bi moralo zadoščati za rešitev naloge iz Preseka.

Če se vam reševanje s programi za simbolično računanje ne zdi prav nič zabavno, programiranje pa vam ni tuje, lahko seveda sami sprogramirate vse potrebne funkcije. No, to je sicer lahko zanimiva in poučna pot, a morda le nekoliko predolgotrajna in prezapletena. Če torej še nimate svoje lastne knjižnice funkcij za delo z velikimi števili (pa tudi če jo imate, a vas skrbi hitrost in zanesljivost delovanja), lahko pobrskate po spletu in uporabite katero od tam ponujenih knjižnic. Primerna naslova za začetek iskanja sta

`directory.google.com/Top/Science/Math/Number_Theory/Software`

in

`primes.utm.edu/links/programs/large_arithmetic`

Ponujene knjižnice so večinoma pripravljene za uporabo iz jezikov C ali C++ (tudi napisane so v teh jezikih, običajno pa je del kode napisan tudi v zbirniku). Če vas ni strah objektov in jezika C++, lahko poskusite s knjižnico NTL (`shoup.net/ntl`). Ljubitelji Linuxa in orodij GNU pa gotovo uporabljate knjižnico GMP (`swox.com/gmp`).

Uspešna uporaba omenjenih knjižnic zahteva solidno znanje jezikov C oz. C++, pa tudi nekaj študija priložene dokumentacije in primerov. Za hitro sprotno preverjanje računov so zato za občasnega uporabnika kar nekoliko nerodne in okorne. Sem pa že pred časom naletel na program UBASIC, ki ga je pred leti napisal matematik prof. Yûji Kida z univerze Rikkyo v Tokiu. Gre za tolmač za programski jezik basic, napisan še za operacijski sistem DOS (kar mu pri srečanju z novejšimi različicami Oken povzroča nekaj težav). Čeprav je že v letih, ima nekaj res lepih lastnosti.

Ena od njih je gotovo ta, da je zastonj. Zadnjo različico, ta ima oznako 8.8f, dobite na naslovu

`ftp://rkmath.rikkyo.ac.jp/pub/ubibm`

(če je gornji naslov nedosegljiv, lahko nekoliko starejšo različico dobite tudi na naslovu

`archives.math.utk.edu/software/msdos/number.theory/ubasic`).

Področje na strežniku vsebuje več datotek. Za nas so še posebej zanimive naslednje: `ub32i88f.zip` vsebuje tolmač `ubibm32.exe` in pomožno datoteko `ubconst7.dat`, dokumentacija programa je v arhivu `ubhelp.zip` (datoteka `ubhelp.xxx`), arhiv `ubiapl96.zip` vsebuje kopico koristnih programov, napisanih v UBASIC-u, `haber.zip` pa je kratek priročnik z osnovnimi navodili za delo z UBASIC-om (tega je pred resnim delom priporočljivo vsaj preleteti). UBASIC pozna tudi grafiko. Nekaj programov, ki uporabljajo grafiko, je zbranih v arhivu `ubgraph.zip`.

Namestitev programa je preprosta. Arhiv `ub32i88f.zip` le razpihnemo v izbrano mapo ter ime mape dodamo v pot. Tolmač zaženemo s klicem `ubibm32` iz ukazne vrstice. Delo z njim končamo z ukazom `system`.

Druga privlačna lastnost UBASIC-a, zaradi katere ga tudi omenjam v tem prispevku, pa je že vgrajeno računanje z velikimi števili. Program sicer pozna več vrst podatkov: poleg treh vrst celih števil še racionalna, realna in kompleksna števila, pa nize, polinome in še kaj.

V nadaljevanju bomo spoznali osnove dela s celimi števili. UBASIC zna računati s števili, ki imajo do 2600 desetiških števk. Osnovni aritmetični operatorji so `+` (seštevanje), `-` (odštevanje), `*` (množenje), `^` (potenciranje), `/` (deljenje, ki da realen rezultat), `\` (celoštevilsko deljenje) in `@` (ostanek pri celoštevilskem deljenju).

Tolmač lahko uporabljamo v sprotnem načinu. Vtipkamo ukaz, pritisnemo Enter in tolmač ga izvede. Npr. `print 2^1000` izpiše nekaj vrstic dolgo veliko celo število. Sprotni način je primeren za hitre, a ne preveč zahtevne izračune ter za spoznavanje delovanja posameznih ukazov.

Pravo programiranje pa običajno poteka tako, da najprej s tekstovnim urejevalnikom (sam uporabljam *TextPad*, v sili pa lahko uporabite tudi *Notepad*) na datoteki s končnico `.ub` pripravimo program v UBASIC-u. Tega nato z ukazom `load "`*ime_datoteke*`"` naložimo v tolmač (uporabimo lahko tudi samo `load` ali pa tipko F1). Če se želimo prepričati, da je pravilno naložen, ga lahko z ukazom `list` (tudi tipka F4) izpišemo na zaslon. Pri tem se oblika programa lahko razlikuje od tiste, v kateri smo ga vtipkali. Program poženemo z ukazom `run` (tipka F5). V primeru težav lahko izvajanje prekinemo s Ctrl+C ali s Ctrl+Break.

Kot zgled si oglejmo program, ki prebere naravno število n, manjše od 10000, ter izračuna in izpiše število deliteljev števila n ter število tistih števil od 1 do n, ki so tuja z n.

```
100  rem Število deliteljev in število tujih števil
110  input "Vpiši število do 10000"; n
120  if or{n < 1, n > 10000} then goto 110
130  delitelji = 0: tuji = 0
140  for i = 1 to n
150    if n @ i = 0 then delitelji += 1
160    if gcd(i, n) = 1 then tuji += 1
170  next i
180  print "Število deliteljev:"; delitelji
190  print "Število tujih števil:"; tuji
200  end
```

Vrstica 100, ki se začne z `rem`, je komentar (komentar lahko začnemo tudi z znakom `'`). Vrstica 120 je primer odločitve s sestavljenim pogojem. Vrstice 140–170 tvorijo zanko `for` s števcem `i`. Funkcija `gcd` (**g**reatest **c**ommon **d**ivisor) iz vrstice 160 vrne največji skupni delitelj dveh števil. UBASIC pozna tudi funkcijo `lcm` (**l**east **c**ommon **m**ultiple), ki vrne najmanjši skupni večkratnik para števil. Tisti, ki veste nekoliko več o teoriji števil, ste gotovo opazili, da je število z n tujih števil od 1 do n, ki jih prešteje gornji program, ravno vrednost Eulerjeve funkcije φ pri argumentu n. To funkcijo pozna tudi UBASIC, in sicer pod imenom `eul` (njen argument mora biti manjši od 2^{32}).

V zadnji številki lanskega letnika Preseka smo iskali najmanjšo potenco števila 2, katere desetiški zapis se začne s številom 2001. Poskusimo z zaporednim preizkušanjem potenc števila 2 rešiti enako nalogo še za število 2002.

```
100  'Najmanjši tak k, da je 2^k = 2002...
110  eksponent = 0: potenca = 1
120  while val(left(str(potenca), 5)) <> 2002
130    eksponent += 1
140    potenca *= 2
150  wend
160  print "2 ^"; eksponent; "="; potenca
170  end
```

Načeloma se lahko zgodi, da je iskani eksponent tako velik, da ima pripadajoča potenca več kot 2600 mest. V takem primeru bi med izvajanjem prišlo do prekoračitve obsega (angl. *overflow*). No, brez skrbi, program po nekaj sekundah računanja najde zapis

$$2^{1652} = 2002\,40922278232540849726515714730105038764536\ldots$$

V programu smo uporabili zanko `while`. Ta se konča pri ukazu `wend`. Funkcija `str` pretvori število v niz, funkcija `left` vrne prvih nekaj znakov niza (v gornjem programu 5, eno mesto porabi predznak), funkcija `val` pa izračuna vrednost, ki jo predstavlja niz.

UBASIC pozna tudi nekaj vgrajenih ukazov za delo s praštevili. Tako klic `prm(`n`)` vrne n-to praštevilo (pri čemer mora biti $n \leq 12251$). Klic `nxtprm(`n`)` vrne prvo praštevilo, ki je večje od n (ukaz zna poiskati le praštevila do 2^{32}). Ukaz `prmdiv(`n`)` pa vrne najmanjši prafaktor števila n (pri čemer ne uspe, če je $n > 2^{34}$ in nima prafaktorja, manjšega od 2^{17}). Prafaktorje manjšega števila izpišemo npr. takole:

```
100  'Razcep na prafaktorje (osnovna varianta)
110  input "Vpisi naravno število"; n
120  while n > 1
130    d = prmdiv(n)
140    print d
150    n = n \ d
160  wend
170  end
```

Za resno preverjanje praštevilskosti pa je treba uporabiti programe iz arhiva `ubiapl96.zip`. Program `prtest1.ub` vsebuje varianto praštevilskega testa Adlemana, Pomerancea in Rumelyja. Datoteka vsebuje kratek glavni program in (nekoliko daljšo) funkcijo `fnADLEMAN(`n`)`, s katero preverjamo praštevilskost. Funkcija je primerna za testiranje števil, ki imajo nekaj deset mest. Izboljšana varianta tega testa je sprogramirana v datoteki `aprt-cle.ub`. Z njo lahko preverjamo tudi števila, ki imajo več kot 300 mest. Mimogrede, po angleško praštevilu pravimo *prime* oz. *prime number*, preverjanju praštevilskosti pa *primality testing*.

Razstavljanje števil na prafaktorje je precej težja naloga kot samo preverjanje praštevilskosti. Razstavljanju sta namenjena programa `ecm.ub` in `mpqsx3.ub` (drugi potrebuje še knjižnico `mpqs#31.ubb`, ki jo samodejno naloži med izvajanjem). S prvim lahko razstavljamo števila, ki imajo več kot 100 mest, a morajo imeti zmerno velike (tja do 20 mest) prafaktorje. V splošnem je čas, ki ga bo porabil postopek, težko vnaprej napovedati. Drugi program je bolj "predvidljiv", a bo zanesljivo zmogel le števila z okoli 50 mesti. Za vajo lahko z enim (ali pa kar z obema) o gornjih programov poiščete razcep števila

$$12345678909876543212345678909876543 21 .$$

Število ima štiri prafaktorje, dva majhna in dva večja.

Martin Juvan